One Time Password Authentication for Open High Performance Computing Environments

April 26, 2004
Stephen Chan sychan@lbl.gov, Stephen Lau slau@lbl.gov,
Jay Srinivasan jsrinivasan@lbl.gov, Adrian Wong atwong@lbl.gov

Abstract

The collaborative nature of Open High Performance Computing (HPC) requires the use of distributed resources that are geographically disparate. Unfortunately, security compromises at individual sites typically results in rapid compromises of other sites that share the same user population. A widespread and troubling vulnerability is the ability of attackers to harvest usernames and passwords using keystroke loggers and modified network servers and clients. One Time Password (OTP) [RFC2289] authentication systems are an important component in dealing with this problem. We examine the requirements for OTP systems in HPC environments, with emphasis on the following:

- End-user usability concerns
- Integration with existing site infrastructure
- The need to support current and emerging applications.
- Guidelines for interoperability between multiple sites operating individual OTP systems.
- Scalability and performance

At a high level, or recommendations can be summarized as follows:

- Rapidly implement a hardware token based OTP system that keeps to a minimum the number of distinct tokens end users must possess
- Migrate Grid credentials to a MyProxy or KCA style service, which is then OTP enabled
- Use Grid credentials or Kerberos to support batch jobs, bulk data transfers and distributed applications.
- Eliminate the use of SSH Keys. When possible, replace them with Grid authentication or Kerberos
- Begin working on tools and procedures to mitigate the security risk from stolen Grid proxies and Kerberos tickets

Motivations

It is not uncommon for different HPC sites to have overlapping user bases. These users typically access these resources from various locations, including home systems and academic and industrial institutions. The distributed and heterogeneous nature of both resources and users presents a tremendous challenge for HPC computer protection. A computer security incident at any of the sites accessed by any user can quickly spread across many institutions, most of which are beyond the administrative control of an individual HPC site.

The common means for user access to HPC sites is via user authentication based on user name and credentials. Although these credentials may vary from site to site, the increased use of "keystroke loggers" and sophisticated network sniffers by attackers poses a great risk to the traditional concept of user name and credentials. Keystroke loggers are hardware or software devices that record keystrokes typed by a user. These devices work on systems regardless if the two end points are using encryption. By using a keystroke logger or network sniffer, an attacker can get access to user, host and password information. If these devices are installed on systems with a large number of users, the threat of widespread compromises becomes apparent.

The ability to install these devices is predicated on an attacker's ability to gain administrator level access on a system. While it may be theoretically possible to constantly keep systems up to date with patches and updates to prevent installation of these devices, the reality is that scheduling problems, resource constraints, organizational priorities and applications requirements often interfere with effectively staying up to date with patches. In addition, it is widely understood that many vulnerabilities are discovered by hackers but not disclosed, therefore there are no security advisories and no patches for these vulnerabilities.

One must assume that any system and user account in a collaborative networked computational environment will fall under attack at some point. The goal is to continually raise the bar to keep attackers off of systems and to minimize the amount of damage if they do gain access to a system.

One Time Password (OTP) authentication is a method to reduce the potential for compromised user credentials. The concept behind OTP is that every session initiated by a user generates a unique user credential that is only valid for that session or for a very short period of time. Even if an attacker is capable of obtaining this user credential, it may either no longer be valid or be prohibited from additional use.

There are many implementations of OTP in existence today [LAM81, CBR03, SECID, SECOMP, CRYPT]. This document highlights the requirements of an HPC deployment of OTP that balances the impact on HPC users while providing an effective deterrent to attackers.

Analysis

There are several key vectors that attackers utilize to gain unauthorized access to HPC sites. This document focuses on user credential compromise. OTP are only part of the solution to this problem – a comprehensive solution includes OTP as part of an overall Best Practices approach for managing systems and cyber security risks. Describing such an approach is beyond the scope of this document. Our purpose here is to discuss specifically how an OTP system can and should be integrated into any existing site security regime, not only to provide better site security at one site but to deter widespread incidents crossing multiple institutions and sites.

It is also beyond the scope of this document to discuss all possible attacks that lead to user credential compromise. We will limit the discussion to specific attacks that are in use and have been seen at HPC sites. In examining the threats we will examine not only the observed threats but also the potential threats that are simple extrapolations of current techniques.

The types of threats that this document addresses are the following.

- Compromised resources that have keystroke loggers and trojaned network services installed to harvest legitimate user credentials.
- Compromise of private user credentials such as SSH private keys and grid certificates by either copying offsite or unauthorized use.
- Forged authorization credentials, such as replacement of existing private keys with the attacker's own keys or certificates.

The immediate and potential threats revolve around replaying user credentials or replaying private key decryption. The well known and technically mature approach to this problem is to use two-factor authentication [CBR03]. This concept relies on something you know (i.e. password), and either something you have (smart card or token) or something you are (biometrics). Smart card/token based approaches are well known within the industry and there is a relatively broad base of talent and experience to draw upon for deployment and management. For the purposes of this document, we focus on token based approaches that do not require specific hardware such as smart card readers or USB interfaces. Adding specific interface hardware requirements would complicate any deployment and also reduce portability – a key consideration given the mobile and geographically diverse user population. While these solutions may have future potential, we do not consider them to be appropriate given the current requirements.

The immediate threat is posed by harvesting of user credentials. Attackers currently appear to be focused on obtaining user names and passwords. User credentials, however, are not primarily user names and passwords. Other credentials include SSH keys and GSI credentials which are used with Grid related resources. Luckily, GSI credentials do not appear to be on the target list of attackers. This may be because most attackers are unaware of Grid related technology. Eventually, one must assume that these attackers will become "Grid-Enabled", especially as Grid technology becomes more mainstream. Therefore any OTP scheme must include the ability to interface with deployed Grid technologies.

GSI proxy credentials and Kerberos [KERB93] tickets are credentials that reside on computers and can be used to access systems without any further authentication. Kerberos has been deployed for roughly two decades and there are few, if any, attacks based on stealing these credentials. One might hope that this trend will continue, and that GSI Proxy certificates will also wallow in protective obscurity. However this is not a safe assumption and steps need to be taken to close down or else mitigate the problems posed by GSI proxies and Kerberos tickets. An important consideration is that Kerberos tickets and GSI proxies are vital to many existing batch processes, and will play an important role in future applications. It is vital that they be integrated into an OTP scheme without sabotaging their batch operation and delegated authentication properties.

Requirements

Unlike most enterprises, OTP deployment in open HPC environments must be able to function in a highly heterogeneous environment. Any deployment must operate on major HPC platforms. These include but are not limited to the following:

- AIX
- Linux (including different distributions)
- Solaris
- HPSS (for storage resources)
- Web
- UNICOS

In addition, as a critical authentication system, any acceptable OTP system must have some form of high availability. Not only does it need to have high availability within a single site, it is also a requirement that network partitions between DOE sites or server outages at remote sites not affect local users. The precise uptime requirements will be dependent on each site's service level agreements, however at a minimum, some form of replication and failover is necessary – as well as the ability for an individual site to continue logging in users, independent of the state of other sites.

A viable OTP deployment must support the following access services:

- SSH (OpenSSH)
- File Transfer (FTP) [FTP85] Protocols (Given the use of ftp, pftp, hsi)
- telnet – telnet has been deprecated in favor of ssh, however in a OTP environment, telnet's clear text password liabilities go away and the lack of encryption aids network intrusion detection systems such as Bro [BRO] and SNORT [SNORT].

A key factor in any type of authorization scheme at HPC sites is the impact on the user base. The ability for users to efficiently access computational, network and storage resources at HPC sites in an unfettered manner is essential. Factors that must be taken into consideration when considering deployment of an OTP scheme at an HPC site include the following:

Ease of deployment.

- User friendliness
- Minimizing interference with high speed bulk data transfers
- Minimizing interference with batch jobs that may need some form of delegated authentication
- Does not interfere with user applications
- Ability to deploy across a wide range of heterogeneous systems
- Ability to be deployable in an environment where the majority of the users are remotely located and highly mobile.

From a user perspective, OTP systems can be an onerous beast. The open HPC environment has unique traits that are often overlooked by developers and vendors of OTP systems. Common complaints from users of deployed OTP systems include the following:

- It is difficult to remember different procedures and Personal Identification Numbers (PINs) used for authentication if multiple sites have separate OTP tokens. Sites that are rarely used end up not being used at all because PIN numbers and procedures are forgotten or lost.
- OTP systems that authenticate connections through a gateway interfere with bulk data transfers.
- OTP systems interfere with batch jobs that perform file transfers.
- Time based passcode OTP systems typically limit users to 1 connection per minute – this can be problematic in situations where multiple login windows need to be opened quickly.

Any successful deployment at HPC sites must take into consideration the traits of Open HPC environments as well as the user experiences from past OTP installations.

Tokens

OTP tokens come in three primary varieties:

1) Tokens that display a time based passcode, updated once a minute
2) Tokens that accept a challenge code as input, and display the response code
3) Tokens that generate a passcode based on some internally measured event.

The first type of token is vulnerable to replay attacks within the span of one minute because the passcode is only updated once per minute. As a result, authentication systems that use these tokens force a 1 minute interval between logins. However, this only applies to logins on a single site – if the same token is used for multiple sites, the passcode could conceivably be replayed. As stated previously, the one minute limit can also severely impede HPC users.

In keeping with the philosophy of asking for something the user has along with something the user knows in order to authenticate them, we believe that OTP deployments should require a tamperproof hardware token. There are "soft" tokens (which are software emulators of the hardware token) which run on PDAs, laptops and browsers, but we feel that the use of these methods will introduce additional vulnerabilities that are best avoided. The essential feature of tokens is that they contain a "secret" known only by the manufacturer and the site that accepts the token. If a software token is used, this secret is either bundled with the software or must be communicated to the user who then types it into the software token. In both cases, the secret is vulnerable to interception, especially if the software token is installed on a machine that is vulnerable to hacking. Hardware tokens do not have these vulnerabilities.

In addition, S/Key [HAL94] and One-Time-Passwords-In-Everything (OPIE) [OPIE95] are systems that use fixed lists of one time passwords. They are conceptually elegant, convenient and inexpensive to deploy, but are infeasible in the large deployments as envisioned in this document due to scalability issues. These systems are essentially "paper token" systems that

have a precomputed list of passcodes that the user carries with them at all times. There are two serious problems which precludes large scale deployment:

- The list of passcodes must be distributed.
  - Because it must be distributed to all users, on a regular basis, it provides opportunities for interception, unauthorized copying and other compromises.
  - In addition, the generation and secure, regular distribution of these lists to the entire user population introduces additional procedural difficulties

- Across multiple sites, it does not seem possible to coordinate passcode use.
  - This opens up the possibility for replay attacks
  - In the long term, passcode lists could easily become far out of synchronization between sites, as heavily used sites burn through passcodes quickly while less used sites have passcode lists that linger

There are software based OPIE and S/Key calculators that simplify the process of managing key lists. They have the same vulnerabilities as the software tokens described above – because of these vulnerabilities; we do not recommend their use.

In order to reduce the number of tokens/key fobs/cards that a user must carry, any OTP should allow one of the following methods to be used for token sharing:

- Allow sites to use the same token/card distributed by another site
  - This approach allows sites to operate independent of each other, while sharing tokens
  - There is a vulnerability in that the secret key for tokens need to be distributed somehow between sites

- Allow cross realm authentication if a local site does not recognize tokens issued by another site
  - This is problematic due to the general issues of trust and reliability across highly distributed systems
  - A network partition or a server outage at a site will prevent all users of that site's tokens from authenticating at other sites.
  - Standards for establishing trust and reliability will need to be established between sites
  - User namespaces would have to somehow be synchronized or else mapped across sites

- Allow tokens/key fobs/cards from multiple vendors to authenticate against the same authentication server.
  - Research done by ORNL [ORNL04] seems to indicate that some form of compatibility is possible across a subset of the token vendors.
  - Further investigation will be needed to evaluate the feasibility of token compatibility.

- Establish a single, centrally managed OTP domain that has replication points at all DOE sites
    - Conceivably a single group, such as ESNet might establish a central OTP service on the ESNet backbone
    - Replication points for the OTP service would be installed at each site
    - Users would only have a single token across all of DOE
    - Logins attempts would be shared across all sites, preventing replay attacks
    - The user namespace management problem also exists in this solution
    - Sites would need to have individual control over system access
        - Possession of a DOE token would not guarantee access to any given site
        - Sites would need to be able to enforce administrative lockouts locally, without effecting remote sites
    - It is unclear if such a system is feasible – further investigation is required

In addition, some OTP systems have a means to issue temporary passcodes in the event that a user loses, misplaces or forgets their token. We consider this to be a valuable feature in a OTP system, but not a hard requirement.

If the same token is used across multiple sites, the following additional requirements must be taken into account:

- Must have some method of ensuring that cross site replay attacks can be prevented
    - Use of challenge response instead of simple time passcode
    - Certify that internally generated event passcodes cannot be replayed in cross site deployments
    - Ensure that different PIN codes are used at different sites
        - Burden on users to remember multiple codes
    - Ensure that authentication services at multiple sites coordinate on throttling logins

We feel that in order to make OTP more palatable for end users, some form of token sharing will be necessary, consequently the cross site requirements must be fulfilled, or a vendor must provide an alternate solution that is not vulnerable to cross-site replay attacks.

Support

OTP systems require support resources, both in deployment and on an ongoing basis for user education and support. Consequently, any OTP system being considered must have a viable long-term solution in terms of support.

One way of ensuring this is to determine if commercial support is available for the product from a company that has a track record of providing acceptable support. In addition, the system must be capable of deployment across organizations of the scale of the DOE laboratories and their associated user communities.

Since the first barrier users typically face in using OTP is in obtaining the necessary hardware, the company should have a token, card or key fob provisioning system that has demonstrably handled a large volume of requests in a reasonably short period of time (as will be expected from various DOE sites).

All sites will have legacy site account management systems in place when deploying OTP; the solution provided should be capable of interfacing with these legacy systems, typically using one or more of the following methods (as appropriate).

- Lightweight Directory Access Protocol (LDAP) [LDAP02] interface
- Active Directory (a Microsoft Directory service) [AD]
- SDK support for C, Java or a scripting language such as Perl.
- Command-line tools so system administrators can implement their own interfaces

Recommended Priority of Deployment

The core concern is to protect the primary mass storage and large scale compute resources at HPC sites. Peripheral services (such as email) are not covered because they are typically not threatened by these exploits due of their limited access methods and existing security measures.

The deployment of OTP to protect compute and storage resources need to be prioritized based on observed threats. The immediate threats are to interactive shell and file transfer logins. Threats that are likely to develop in the future are attacks on SSH keys, Grid credentials and Kerberos. As a result, we recommend that the deployments deal with problems in the following phases:

1) Eliminate the ability to replay shell and file transfer logins
2) Eliminate the ability to replay SSH key and GSI private key decryption
3) Integrate Kerberos tickets and GSI proxies into an OTP scheme and place careful controls on their lifetime and the ability to copy them to unauthorized locations.

Requirements that OTP should satisfy in addressing these three concerns are outlined below along with some explanation of the rationale for those requirements. We provide recommendations for each of tasks, as well as requirements that need to be fulfilled by a solution. At a high level, we recommend that an OTP solution be deployed, and that an OTP enabled MyProxy or Kerberos service be layered on top to provide support for batch jobs, bulk file transfers and distributed applications. A properly extended MyProxy or Kerberos service is a viable solution to some of the shortcomings in past OTP implementations. On the flip side, the use of GSI and Kerberos credentials without proper administrative controls has the potential to be a major security liability.

Eliminate the Ability to Replay Shell and File Transfer Logins

The keystroke logging and sniffing attacks have been most successful in compromising systems by capturing usernames and credentials during shell logins and file transfers. These attacks allow hackers to access legitimate accounts on remote systems, and result in compromises spreading rapidly. Consequently, this vulnerability must be eliminated rapidly and effectively.

However, OTP systems that protect shell and file transfer access will undoubtedly affect other uses of the systems as well, such as batch processing, bulk file transfers, and possibly interfere with the operation of distributed applications. Based on past experience with OTP systems and consideration of user applications, we feel that a viable OTP system for Open HPC must fulfill these requirements:

- Does not impede the use of bulk data transfers at and between sites
- Must not preclude batch jobs
- Must not preclude communication amongst the components of a distributed application (such as those using the Grid).
    - Applications must have legitimate credentials from GSI or, Kerberos or a similar authentication mechanism.

For bulk data transfers, batch jobs that need their own authenticated connections and processes that need non-interactive authentication, we recommend that services be Globus GSI enabled and that a Globus proxy-only service or an equivalent Kerberos service be used. The proxy or ticket should of course be issued using the OTP mechanism and have a fixed and short lifetime.

Integrating OTP in an environment with minimal disruption to user's access methods also requires that the OTP mechanism must be supported by common clients used. This can be done via one of the following methods:

- Pluggable Authentication Modules (PAM) [PAM95] PAM modules for all listed platforms and services
- The Remote Authentication Dial-In User (RADIUS) [RFC2138] Service, protocol both of which will allow users to continue accessing the system using the same methods as they are currently using.

PAM is a widespread standard for modularizing authentication on Unix and Windows platforms. Using PAM, different forms of authentication, including OTP, can be "plugged into" an existing service. To ease the transition to OTP, we require the OTP system to have some form of PAM interface.

RADIUS is popular among network hardware devices such as dial-up modems. It is also a protocol that is well supported by PAM – for platforms that do not directly support PAM, a RADIUS interface can often be used.

Eliminate the Ability to Replay SSH Key and GSI Private Key Decryption

A longer term danger is in the ability of keystroke loggers and similar tools to compromise SSH keys and Globus certificates. While this has not yet been observed in actual compromises, it is a simple extrapolation of existing hacker tools. Control of SSH keys and Globus certificates allow hackers to impersonate a legitimate user and canceling a compromised SSH key or Globus certificate can be problematic due to the lack of centralized administration.

Possible methods to secure SSH private key include the following:

- Force SSH private keys to require a second authentication (typically using OTP).
  - This essentially makes private keys useless, and will require some user education before it can be enforced.
- Disable the use of private SSH keys entirely. Again, this is a step that must be taken with considerable prior user education.
- Convert to using a Globus-based SSH mechanism and use an OTP method to generate the GSI proxy-certificates. This step will again require user education, as well as setting up the entire infrastructure to generate GSI proxy-certificates and manage them securely (part of NSF Middleware Initiative – NMI).

When using GSI certificates, sites can improve the security of the certificates issued by one of several methods:

- Place all long-term certificates in a central credential store (such as MyProxy) [NOV01] which has additional OTP authentication.
- Eliminate long-term certificates:
  - Use a Kerberos based (KCA/KX509 type) service [DOS01, KOR01] that requires OTP before ticket validation..
  - Use a proxy-based service that supports OTP directly using PAM or RADIUS.

MyProxy is a server that is capable of storing and signing credentials for a user. Typically it is only used to store short term certificates for the user, and a user is personally responsible for managing the long term Globus credential. We recommend that long term Globus credentials be removed from cluster systems, laptops and other highly vulnerable locations, and managed centrally on a MyProxy server. There is active work being done on integrating MyProxy with OTP solutions via PAM.

KCA/KX509 are tools that convert Kerberos credentials into short term Globus credentials. They are popular and useful for sites that already have a mature Kerberos installation. If a site is using Kerberos, we recommend integrating OTP with Kerberos and using either Kerberized services or Globus services using KCA generated credentials.

Integrate Kerberos Tickets and GSI Proxies into an OTP System and Place Controls on Usage

This is a much more difficult task when compared to the previous tasks because satisfactory solutions do not yet exist. It is also seems to be the least immediate concern when based upon operational experience. However, there are no apparent technical obstacles to hackers exploiting vulnerabilities in Kerberos tickets and GSI proxy certificates.

Kerberos tickets and GSI proxy certificates are both short term credentials stored in local machines that are used for authentication. Possession of a valid Kerberos ticket or GSI proxy typically bypasses login procedures such as prompting for username and password. This is actually a valuable property of these credentials, because it allows bulk data transfers, batch jobs and distributed applications to operate without constant user intervention. However, this will need to be integrated somehow into an OTP regime.

The following recommendations can mitigate the danger from compromised Kerberos and GSI proxy credentials:

- Only issue proxy certificate or Kerberos ticket if the user authenticates to a MyProxy or Kerberos service that uses OTP
- Limit services an individual proxy/ticket can be used on
- Limit sites that a proxy/ticket can be used to and from
- Place strict limits on the proxy/ticket lifetime
- Deploy an Online Certificate Status Protocol (OCSP) [RFC2560] service to notify remote sites when a ticket or proxy has been cancelled.

Summary

The introduction of One Time Passwords into a site infrastructure necessarily involves many changes; however it seems to be the best long term option to secure legitimate accounts from being compromised by replay attacks. Based on that observation, we have attempted to carefully evaluate the consequences of OTP on users, applications and how such a system can be reasonably deployed within an existing site.

We looked at the problems with OTP systems themselves: the various forms of tokens, the potential for shared tokens to bypass replay safeguards, the limitations of S/KEY and OPIE, and the potential for incompatible OTP implementations. We recommended tokens and methods that we felt were appropriate to a large scale, cross site deployment.

A key consideration is how does OTP affect the user community and their applications? Computer clusters are operated on behalf of users and their applications and it is vital that any OTP system be handled in a way that minimizes the burden on users and does not interfere with their applications. Based on past experience, our recommendations attempt to minimize the number of tokens, PIN numbers and distinct procedures that users need to remember. In addition, we offer a method of applying OTP to batch processes and distributed applications.

From the site administrator's point of view, there are many challenges that need to be faced when deploying such a system. OTP systems have an internal database of users. The most basic hurdle is integrating this database with an existing site user database. By requiring API's or automatic synchronization with LDAP, we address this database issue. By specifying PAM and RADIUS support, we address how existing HPC resources can communicate to the OTP service. We expect that a vendor be able to provide professional technical support and also be capable of

providing the hardware tokens in a timely fashion. In some instances, existing services, such as private key SSH authentication, will have to be shut down or heavily modified. Where possible we want to offer an approach that offers the same functionality, but is less vulnerable to password vulnerabilities.

There is also the problem of how OTP interacts with distributed computing. Many distributed computing systems have the notion of a delegated identity, either through Globus proxies or Kerberos tickets. We suggest methods of integration with OTP, some of which are already in progress (MyProxy), as well as methods of mitigation when tickets and proxies are risk of being stolen. In particular, we recommend that long term Globus credentials be removed from cluster systems, laptops and other highly vulnerable locations, and managed centrally on a MyProxy server. This is probably the most feasible, and important action that will improve the situation.

Finally, we presented a prioritization of the work that needs to be done. OTP has very broad ranging effects and it is important that the most pressing issues be dealt with first. In addition, there is technology that needs to be developed and deployed – we identified the work that we feel needs to be done, and prioritized it based on current observations.

In writing this document, we have tried to address the issues in a general fashion, and have left out many details that didn't relate to the general problem. We have included as much detail in the recommendations section as we felt appropriate to address the general problem, and purposely left out many details that depend on site specific information. Much more investigation will be necessary to arrive at an implementation for a given site, but we hope that this document has served as an introduction to the issues and consequences of OTP implementation.

References

[AD] http://www.microsoft.com/technet/prodtechnol/windows2000serv/technologies/activedirectory/deploy/projplan/adarch.mspx

[BRO] Paxson, V., *Bro: A System For Detecting Network Intruders in Real Time, Computer Networks,* 31(23--24), pp. 2435-2463, Dec. 1999

[CBR03] Cheswick, W. R., Bellovin, S. M., Rubin, A. D., *Firewalls and Internet Security: Repelling the Wily Hacker*, Addison-Wesley, 2003. and references therein.

[CRYPT] http://www.cryptocard.com/

[DOS01] Doster, W., Watts, M., Hyde, D., *The KX.509 Protocol,* February 2001. http://www.citi.umich.edu/techreports/reports/citi-tr-01-2.pdf

[FTP85] Postel, J., Reynolds, J., *RFC 959: File Transfer Protocol (FTP).* http://www.ietf.org/rfc/rfc0959.txt

[GSI04] Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S., Seibenlist, F., X.509 Proxy Certificates for Dynamic Delegation. http://www.globus.org/Security/papers/pki04-welch-proxy-cert-final.pdf

[HAL94] Haller, N. M., *The S/KEY One-time Password System, In the Proceedings of the Internet Society Symposium on Network and Distributed System Security,* San Diego, Feb 3, 1994.

[KERB93] Kohl, J., Neuman, B. C., *RFC 1510: The Kerberos Network Authentication Service (Version 5*). http://www.ietf.org/rfc/rfc1510.txt

[KOR01] Kornievskaia, O., Honeyman, P., Doster, W., Coffman, K., *Kerberized Credential Translation: A Solution to Web Access Control*, February 2001. http://www.citi.umich.edu/techreports/reports/citi-tr-01-5.pdf

[LAM81] Lamport, L. *Password Authentication with insecure communication, Communications of the ACM*, 24(11):770-722, Nov. 1981

[LDAP02] Hodges, J., Morgan, R., *RFC 3377: Lightweight Directory Access Protocol (v3): Technical Specification.* http://www.ietf.org/rfc/rfc3377.txt and references therein.

[NOV01] Novotny, J., Tuecke, S., Welch, V., *An Online Credential Repository for the Grid: MyProxy. In the Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10)*, IEEE Press, August 2001.

[OPIE95] McDonald, D. L., Metz, C., *One Time Passwords in Everything (OPIE): Experiences with Building and Using Stronger Authentication. In the Proceedings of the 5th USENIX Security Symposium*, Salt Lake City, June 1995.

[ORNL04] Private communication from ORNL.

[PAM95] Samar, V., Schemers, R., *Unified Login with Pluggable Authentication Modules (PAM)*. http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt

[RFC2138] Rigney, C., Rubens, A., Simpson, W., Willens, S., *RFC 2138: Remote Authentication Dial In User Service (RADIUS)*. http://www.ietf.org/rfc/rfc2138.txt

[RFC2289] Haller, N., Metz, C., Nesser, P., Straw, M. *RFC 2289: A One-Time Password System*. http://www.ietf.org/rfc/rfc2289.txt

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C., *RFC 2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP*. http://www.ietf.org/rfc/rfc2560.txt

[SECCOMP] http://www.securecomputing.com

[SECID] http://www.securid.com

[SNORT] http://www.snort.org